Implementasi Pengenalan Multiple Marker untuk Sistem Augmented Reality

Steven Febrian[1], Liliana[2], Kartika Gunadi[3]
Program Studi Teknik Informatika, Fakultas Teknologi Industri, Universitas Kristen Petra
Jl. Siwalankerto 121-131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) - 8417658
E-mail: stevenfebrian1@gmail.com[1], lilian@petra.ac.id[2], kgunadi@petra.ac.id[3]

ABSTRAK

Pembuatan maket untuk pencontohan kepada para investor terkadang memerlukan biaya yang sangat besar, pembuatan maket sering berganti-ganti jika dirasa ada yang kurang cocok dalam pembuatannya. Untuk mengatasi ketidak cocokan antara *user* dan investor, para desainer akan membuat ulang maket-maket tersebut jika ternyata tidak sesuai dengan keinginan oleh investor. Akan tetapi masalah yang timbul adalah pembelian alat contoh peraga yang mahal, sehingga tidak efisien dalam menghasilkan pembuatan biaya maket.

Untuk mengatasi hal ini , dibuatlah aplikasi visualisasi maket taman kota dengan menggunakan marker yang terbuat dari kertas, sehingga biaya pembuatan maket tersebut murah, dibandingkan dengan uang yang harus dikeluarkan untuk membeli alat-alat seperti, pohon, bangku taman, dll. *User* juga dapat menambahkan maket sesuai dengan keingginannya, dapat menambahkan object yang terdapat *free* di internet, lalu memilih marker sesuai dengan keinginannya, bahkan *user* dapat mencetak sendiri maket yang akan digunakannya.

Untuk membuat visualisasi maket tersebut dilakukan proses-proses yang cukup panjang mulai dari merubah warna menjadi keabuan, merubah warna menjadi hitam dan putih, deteksi tepi, mendeteksi persegi, mencocokan *template* hingga mencetak *object* 3 dimensi yang diinginkan.Melalui hasil pengujian, telah didapatkan hasil yaitu *object* yang tercetak pada layar sudah sesuai posisinya dengan *marker*. Kelemahan dalam proses ini adalah proses rendering yang memakan cukup lama dan juga proses pencetakan object 3D yang masih belum memiliki *texture* dan *marker* yang dikenali harus berbentuk persegi.

Kata Kunci: *Augmented Reality*,Visualisasi,*marker detection,marker recognition,virtual reality.*

ABSTRACT

Making mockups for display to investors often requires a very high cost, manufacture mock floater if deemed there are less suitable in the making. To overcome this age often the designers will re-create the mock-scale model if it is not in accordance with the wishes of investors. But the problem that arises is the purchase of an expensive tool visual examples, so it is not efficient in generating cost manufacturing maket.

To overcome this, invented a visualization application mockups city park by using markers made of paper, so that the cost of making these mockups cheap, compared with money must be spent to buy equipment such as, trees, park benches, etc. Users can also add mockups according to keingginannya, can add objects that are free on the Internet, then choose a marker in accordance with his wishes, the User can print the even own mockups that will be used.

To make the visualization process is carried out mock-long process ranging from changing color to gray, change the color to black and white, edge detection, square detecting, matching the template to print the object desired dimensions. Through the test results, it was found that the result object that is displayed on the screen is in conformity with the position marker, the downside of this process is that the process that takes a long time rendering and 3D printing process object that still has texture and recognizable marker must rectangular.

Keywords: *Augmented Reality,Visualisation,marker detection,marker recognition,virtual reality.*

1. PENDAHULUAN

Augmented reality adalah teknologi yang menggabungkan benda maya dua dimensi dan ataupun tiga dimensi ke dalam sebuah lingkungan nyata tiga dimensi lalu memproyeksikan benda-benda tersebut dalam waktu nyata. Tidak seperti realitas maya yang sepenuhnya menggantikan kenyataan, namun *Augmented Reality* hanya menambahkan atau melengkapi kenyataan [2].

Sistem *Augmented Reality* bekerja berdasarkan deteksi citra dan citra yang digunakan adalah marker. Prinsip kerjanya sebenarnya cukup sederhana. kamera yang telah dikalibrasi akan mendeteksi *marker* yang diberikan, kemudian setelah mengenali dan menandai pola *marker*, webcam akan melakukan perhitungan apakah marker sesuai dengan database yang dimiliki. Bila tidak, maka informasi *marker* tidak akan diolah, tetapi bila sesuai maka informasi marker akan digunakan untuk me-render dan menampilkan objek 3D atau animasi yang telah dibuat sebelumnya.

Taman kota adalah salah satu sarana yang digunakan baik oleh para remaja, anak-anak bahkan orang tua sekalipun untuk mencari sarana hiburan dan sekedar untuk menikmati waktu luang, sayang sekali penataan taman yang kurang baik terkadang membuat orang lain tidak suka berlama-lama di taman bermain tersebut. Aplikasi ini dapat membantu pengembang pembuat taman bermain untuk menentukan lokasi-lokasi yang tepat untuk menaruh kursi, bunga ataupun tempat bermain yang baik, sehingga dapat membuat taman menjadi jauh lebih baik dan dapat dinikmati dengan baik oleh pengunjung. Maket adalah suatu tambahan atas rancangan arsitektur dan sebagai cara utama untuk menyampaikan ide dan menggambar tata ruang. Motivasi

membuat maket adalah memungkinkan perancang untuk menguji kualitas rancangan dalam skala kecil dan membantu perancang dalam mengembangkan sentuhan atas ruang, estetika, dan bahan. Sebuah maket membantu para perancang untuk mendemonstrasikan bakat dan kualitas mereka dalam hal ide proyek. Biaya dalam pembuatan maket dapat dikatakan masih cukup mahal, dikarenakan komponen-komponennya yang banyak dan berbentuk sangat kecil, dan terkadang kita harus mengganti komponen dikarenakan ketidak cocokan sebuah komponen dengan komponen-komponen yang lainnya, komponen dalam hal ini dapat berupa, tanaman, pohon, bangku taman, bunga, lampu, dll.

2. TEORI PENUNJANG

2.1. *Metode Grayscale*

Grayscale adalah metode untuk merubah format image gambar yang berawal dari 3 channel yaitu red, green dan blue menjadi 1 channel saja yaitu gray. Rumus dalam merubah gambar menjadi gray adalah[2].

S=((0.299*R)+(0.586*G)+(0.114*B)). (1)

2.2. *Thresholding*

Threshold adalah metode segmentasi sederhana intensitas [4]. Untuk membedakan piksel, dilakukan perbandingan dari setiap nilai intensitas piksel dengan ambang batas yang ditentukan. Setelah itu, nilai suatu piksel bisa diberi warna hitam atau putih tergantung dari perbandingan nilai piksel itu dengan ambang batas.

2.2.1 Threshold Binary

Rumus dalam *threshold binary* adalah:

$$dst\,(x,y) = \begin{cases} 255 & if\ src(x,y) > batas\ threshold \\ 0 & otherwise \end{cases} \quad (2)$$

src adalah *image* awal, *dst* adalah *image* hasil *threshold.* Jadi, jika intensitas piksel lebih tinggi dari nilai batas *threshold* maka intensitas piksel baru diganti dengan *maxvalue* dan sebaliknya.

.

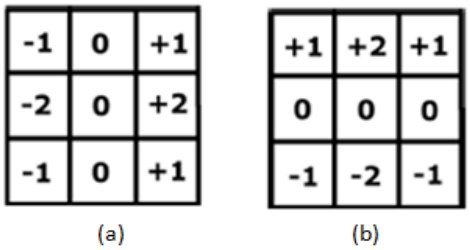
2.2.2 Threshold Adaptive

Pada *adaptive threshold*, suatu citra dibagi menjadi blok-blok kecil dan kemudian dilakukan *threshold* lokal atas setiap blok itu dengan nilai ambang batas yang berbeda. Nilai ambangnya ditentukan sebagai fungsi rata-rata nilai piksel di dalam daerah citra tersebut. Metode yang digunakan adalah setiap piksel akan di-*threshold* dengan ambang batas nilai yang dihitung dari nilai rata-rata piksel tetangga. [2],[5].

2.3. *Edge detection*

Canny Edge Detector merupakan salah satu metode yang paling umum digunakan dalam mendeteksi tepi untuk pengolahan citra [1]. Langkah-langkah untuk *melakukan canny edge detector*:

1. Pengurangan *noise*
   Pengurangan *noise* dapat dilakukan dengan menggunakan metode *gaussian filter*.
2. Menghitung panjang *gradien* dan besar sudut *gradient*



Gambar 1. (a) Kernel Dx (b) Kernel Dy

Rumus menghitung besar gradient adalah:

$$\boldsymbol{D = \sqrt{Dx(x,y)^2 + Dy(x,y)^2}} \quad (3)$$

Dx adalah hasil perkalian antara piksel gambar dengan kernel pada Gambar 1(a) dibagi dengan jumlah kernel. Dy adalah hasil perkalian antara piksel gambar dengan kernel pada Gambar 1(b) dibagi dengan jumlah kernel[3].

Rumus untuk menghitung besar sudut *gradien* adalah:

$$\theta = \arctan(\frac{Dx(x,y)}{Dy(x,y)}) \quad (4)$$

3. *Non-Maximum Surpression*
   Non-Maximum Surpression menyatakan suatu piksel sebagai tepi (*edge*) apabila piksel tersebut memiliki besar *gradien* yang terbesar. Jika pixel (x, y) memiliki *gradien* yang paling besar dari semua piksel yang diperiksa, maka piksel tersebut merupakan tepi (*edge*)[4].
4. *Hysteresis Thresholding*
   Hysteresis Thresholding memiliki dua ambang batas, ambang batas bawah (*tlow*) dan ambang batas atas (*thigh*). Jika piksel (x, y) memiliki *gradien* kurang dari *tlow*, maka piksel tersebut bukan merupakan tepi (*edge*). Jika piksel (x, y) memiliki *gradien* lebih besar dari *thigh*, maka piksel tersebut merupakan tepi (*edge*). Jika piksel (x, y) memiliki *gradien* antara *tlow* dan *thigh* dan salah satu dari piksel sekeliling di daerah 3 × 3 memiliki *gradien* lebih besar dari *thigh*, maka piksel tersebut merupakan tepi (*edge*)[1].

2.4 *Rectangle Detection*

Diasumsikan bahwa marker yang digunakan adalah marker berbentuk persegi maka yang dilakukan pertama kali adalah dengan mencari bagian mana dari gambar tersebut yang berbentuk persegi, metode yang digunakan adalah dengan mencari titik potong antar 2 garis dengan menggunakan persamaan sebagai berikut

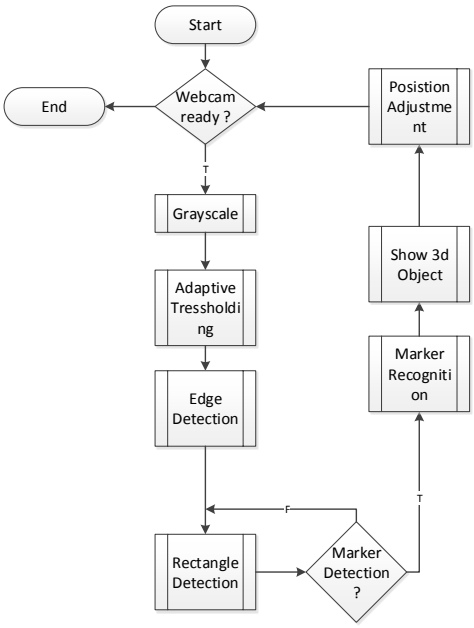
$$\frac{y-y1}{y2-y1} = \frac{x-x1}{x2-x1} \quad (5)$$

Persamaan garis yang terbentuk akan menyerupai ax+by+c = 0.

3. DESAIN SISTEM

Rancangan arsitektur dan proses kerja sistem secara garis besar digambarkan dalam pada Gambar 2 dan Gambar 3. Pertama *User* diminta untuk memasang *webcam* luar, menyiapkan kertas, mengatur posisi *webcam* dan kertas serta menekan tombol *start*, kemudian dilakukan proses *grayscale, tressholding*, lalu dilanjutkan dengan proses *edge detection*, *rectangle detection*, marker *detection* lalu marker *recognition* dan terakhir adalah menampilkan *object* pada layar.

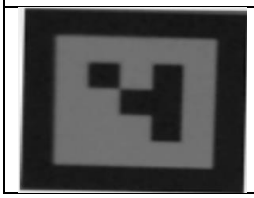
Gambar 2. Arsitektur secara garis besar



Gambar 3. *Flowchart* cara kerja program

Pada proses *template matching* proses yang dilakukan adalah proses *matching block by block*, dimana pada setiap gambar akan dilakukan *resize* gambar menjadi berukuran 70x70 *pixel*, dan setelah itu proses yang dilakukan adalah membagi gambar tersebut menjadi *block-block* berukuran 10 *pixel* sehingga pada setiap gambar akan mempunyai kode berupa *array 2d* berukuran 7x7 lalu . Setiap gambar akan dilakukan proses rotasi sebanyak 4x untuk melakukan pengecekan apakah marker tersebut mempunyai kecocokan dengan gambar atau tidak seperti pada tabel 1.

Tabel 1. Proses *resize* gambar dengan *threshold*

Gambar *camera*	Gambar setelah resize dan *threshold*
	

Pada percobaan tabel 1 dapat dilihat pada gambar *camera* adalah gambar yang didapat dari capture camera, pada proses selanjutnya gambar tersebut akan diproses yaitu di *resize* menjadi ukuran 70x70 dan di *threshold* sehingga gambar akan memiliki 2 nilai saja yaitu 0 atau 1.

Setelah gambar *dithreshold* gambar akan dibagi menjadi 7x7 dan akan menghasilkan nilai sebagai berikut, hasil *block by block recognition* dapat dilihat pada tabel 2.

Tabel 2. Hasil pengujian *block by block recognition*

Gambar *threshold*	Hasil *recognition*
	1111111 1000001 1010101 1011001 1010001 1111111

Pada tabel 2 hasil pengujian *block by block recognition* adalah hasil percobaan dari gambar yang telah di *threshold* lalu dilakukan proses *recognition* yaitu mengidentifikasikan gambar tersebut secara 10x10 *pixel* sehingga total nilai yang dimiliki adalah 49 nilai yang mempunyai 7 baris dan 7 kolom yang selanjutnya hasil proses recognition yang berupa *array* 2d akan diocokan dengan *database* yang telah dimiliki sebelumnya yaitu berupa *text file* yang akan dibaca pada awal program.

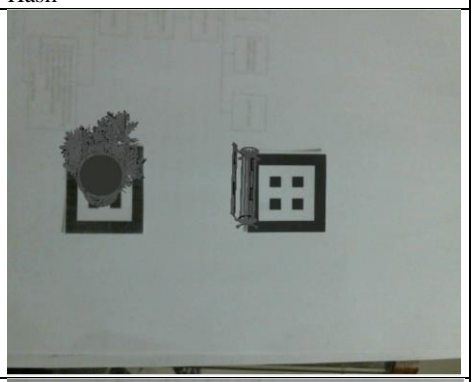
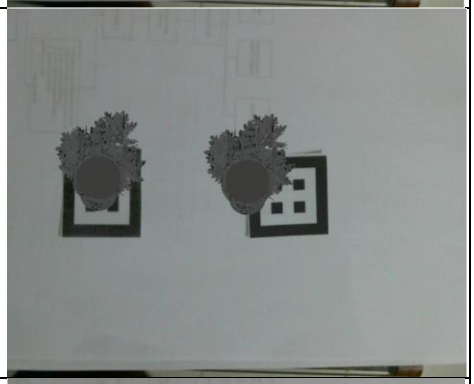
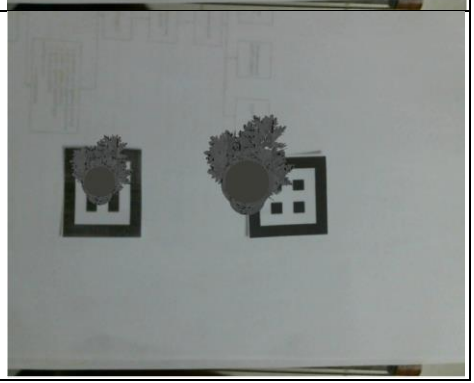
Hasil pencocokan tidak akan selalu tepat 100% melainkan pengguna dapat memasukan sendiri batas toleransi yang dikehendaki dengan rentang 0-100% dengan asumsi ketika 0% maka semua gambar akan dianggap sama sedangkan ketika 100% maka gambar harus sama tepat. Jika toleransi sebesar 70% maka dari 49 nilai yang dicocokan harus memiliki kecocokan sebesar minimal 34 *pixel* dan ketika toleransi yang diberikan sebesar 90% maka minimal harus memiliki 44 *pixel* yang cocok.

4. PENGUJIAN

Pengujian terhadap aplikasi dibagi menjadi beberapa bagian diantaranya adalah sebagai berikut:

- Pengujian terhadap pengenalan marker dengan menggunakan *block by block recognition* dengan toleransi tertentu.
- Pengujian pembuatan maket dengan *terrain* dan tanpa mengunakan *terrain.*
- Pengujian dengan tingkat toleransi tertentu dapat dilihat pada tabel 3.

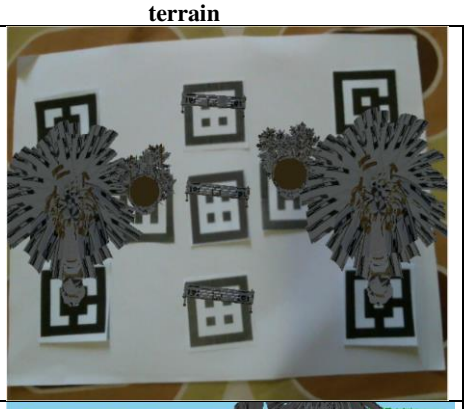
Tabel 3. Hasil percobaan dengan toleransi

Toleransi	Hasil
100	
95	
70	
40	

Pada proses yang terlihat pada tabel 3 bahwa pada toleransi diatas 95% 2 buah marker tersebut menunjukan object yang berbeda tetapi pada toleransi dibawah 70% maka object yang terlihat sama, hal ini disebabkan karena marker yang tidak terlalu jauh berbeda sehingga ketika batas toleransinya hanya 70% 2 buah marker tersebut akan dianggap marker yang sama.

Pengujian pembuatan maket dengan mengunakan terrain dan tanpa menggunakan terrain dapat dilihat pada tabel 4.

Tabel 4. Pengujian dengan menggunakan terrain dan tanpa terrain

Tanpa *terrain*	
Dengan *terrain*	

pada tabel 4 terdapat 2 hasil pengujian yaitu yang pertama adalah pengujian tanpa menggunakan *terrain* dimana gambar yang ditampilkan adalah gambar asli dari *camera* tanpa menambahkan sesuatu sebagai *layout*,sedangkan gambar ke 2 adalah gambar pengujian yang dilakukan dengan memberikan terrain sehingga terlihat seperti memiliki latar belakang dan *layout* yang berwarna hijau menyerupai rumput di padang rumput dan biru adalah sungai.

5. KESIMPULAN

Berdasarkan hasil pengujian didapatkan beberapa kesimpulan yaitu:

- Pendeteksian marker harus menggunakan marker yang berbentuk persegi, apabila bentuk marker tidak berbentuk persegi maka marker tidak akan dikenali. Proses pengenalan marker tidak tergantung kepada intensitas cahaya, jika cahaya yang digunakan berlebih selama tidak mempengaruhi proses *threshold* dan tidak ada bagian marker yang terpotong maka tidak akan menjadi masalah.
- Pengunaan *camera* dengan resolusi tinggi menyebabkan proses pengenalan marker lebih lama,tetapi mempunyai ketepatan yang lebih baik hal ini dikarenakan dengan menggunakan proses *camera* yang mempunyai resolusi lebih rendah maka proses pengecekan akan berjalan lebih cepat karena *pixel* yang lebih sedikit dari pada menggunakan *camera* dengan resolusi tinggi yang otomatis membuat *pixel* lebih banyak.
- Jumlah marker yang terdeteksi dapat sebanyak mungkin hanya saja proses pengenalannya akan menjadi sangat lambat karena banyak marker yang harus dimatchingkan dengan gambar yang ada

- Proses *matching* yang digunakan adalah *block by block recognition*, proses tersebut dapat berjalan dengan cukup baik dan dapat mengartikan marker . Permasalahan yang terjadi adalah ketika marker yang digunakan mempunyai bentuk yang simetris baik itu sumbu y maupun sumbu x, karena ketika gambar tersebut mempunyai bentuk yang simetris proses rotasi tidak akan dapat berjalan dengan sempurkan karena ketika gambar berotasi sebanyak 90,180,270,360 tetap mempunyai kode yang sama.

6. DAFTAR PUSTAKA

[1] Akhmad Afissunani, Akuwan saleh, M. Hasbi Assidiqi.2014. *Multi Marker Augmented Reality untuk Aplikasi Magic Book.*

[2] *Kadir,Abdul.2013. Dasar Pengolahan Citra Digital dengan Delphi. Yogyakarta: Penerbit Andi.*

[3] Li, Z., Wong, K.H., Gong, Y. dan Chang M.Y. 2011. *A Method for Movable Projector Keystone Correction*. IEEE Transaction on Multimedia, Vol. 13, No. 1, February 2011.

[4] Surampalli, G.P. 2012. *An Analysis of Skin Pixel Detection using Diffrenet Skin Color Extracction Techniques*. International Journal of Computer Applications, Vol. 54, No. 17, September 2012.

[5] Zhou, P., Ye, W., Xia Y., & Wang, Q. 2011. *An Improved Canny Algorithm for Edge Detection. Journal of Computional Information Systems*, 7(5): 1516-1523, May 2011.